Destroying the WSRF Resource (Immediate Destruction)

Version: 1.0 Date 11/07/05

This tutorial is the continuation of fourth tutorial "HelloWorld WSRF with multiple instances" which can be found at http://tyne.dl.ac.uk/WOSE/Tutorial4.pdf. In last tutorial we had very simple "WSRF HelloWorld" example using Factory Design Pattern, in which for each client there was a separate instance of resource, when ever a new instance was created unique identifier was sent to client along with the location of Service to access it. Still two clients can interact with single instance, if they both use same unique identifier. In this tutorial client can delete the instance of the resource. This tutorial is very simple and it is not complicated as compared to last tutorial and still we are using the Factory Design Pattern.

Most of the deployment Descriptors and java classes for this tutorial are same and can be used as they are. There is one important thing, which creates bit confusion, you have to change the name of classes for the Resource and ResourceHome for each new service, for two services with the same name for Resource and ResourceHome, recently deployed Resource and ResourceHome will be used. What I mean is if two Services S1 and S2 are deployed and each of them have same class names for the Resource and Resource Home i.e. xxx.xxx.xxx.R and xxx.xxx.xxx.RH, then this will confuse the container. If S2 is deployed after S1, then xxx.xxx.xxx.R and xxx.xxx.xxx.RH for S2 will be used, even when you are calling S1. I had the illusion that similar name will not conflict as it is the case for the Servlets as each of them have their on directory structure, but that is not true for WS Core and problem can be due to the way Axis handle Web Services (I am not sure who is exactly culprit as this is not possible for JAX-RPC Web Services).

Now we have the following classes

1. HelloQNames.java; it is same interface with QNames, only Namespace has changed.
2. HelloWorldFactoryService5.java; this is new class with createResource(), which is responsible to create instance of the resource with unique ID and sending the URL of HelloWorldService to the client.
3. HelloWorldResourceFactory5.java; it is same as HelloWorldResourceFactory.java, representing the resources for our WSRF HelloWorld.
4. HelloWorldResourceHomeFactory5.java; this is home for our resource and resource has to be initialised through this class, in previous tutorial it was HelloWorldResourceHomeFactory.java
5. HelloWorldService5.java; this is the class which has business methods for our main service, in last tutorials it was HelloWorldService.java.

Destroying the WSRF with multiple instances

We will start with simple WSRF Service, which, you have already seen in the last tutorials and discussing the changes which you need to made in the tutorials. Our simple WSRF *Service*, had two business methods, echo() and getNameRP(), and one compulsory method GetResourceProperty() and then we had added new business method called create() (in Tutorial 3, but in Tutorial 4 it is createResource()), both in the WSDL file and in java implementation. Client can't interact with any business method of our web service unless first he calls the create()/createResource(), in return operation will create new instance of resource for "WSRF HelloWorld" and will return the unique identifier to the client. Client will use this unique identifier for further interaction with the web service. In this tutorial we will add a new method destroy, which will let the client to destroy the instance of the resource, luckily Globus implementation provides the implementation of this operation, so we don't need to implement it but we have to add this method in the WSDL file for HelloWorldService.

Modifying WSDL File:

Just like previous tutorials we are starting with our WSDL file; we have two WSDL files one for HelloWorldService and one for HelloFactoryService. WSDL file for HelloFactoryService is "factory5_port_type.wsdl" and is exactly the same as last tutorial; the only change is the change in the target name space. There are in total 4 changes highlighted by bold blue font.

Destroying the WSRF Resource (Immediate Destruction)

These WSDL files are in the directory “schema\tutorial” relative to tutorial “src” directory with the name “helloworld5_port_type.wsdl” and “factory5_port_type.wsdl.

We have to change the WSDL file for HelloWorldService which is helloworld5_port_type.wsdl, other than changing the target name space; we have to add the Destroy operation. We are using the operation provided by the Globus implementation, therefore we don’t have to add any data type or input/output messages, just have to add the operation in the element portType .

```
<operation name="Destroy">
  <input message="wsrlw:DestroyRequest"
   wsa:Action="http://.../wsrf-WS-ResourceLifetime/Destroy"/>
  <output message="wsrlw:DestroyResponse"
   wsa:Action="http://.../wsrf-WS-ResourceLifetime/DestroyResponse"/>
  <fault
     message="wsrlw:ResourceNotDestroyedFault"
     name="ResourceNotDestroyedFault"/>
  <fault message="wsrlw:ResourceUnknownFault"
     name="ResourceUnknownFault"/>
</operation>
```

The complete WSDL file helloworld5_port_type.wsdl is shown below with changes in bold blue font which is similar to helloworld4_port_type.wsdl except the addition of Destroy Operation:

```
<definitions name="HelloWorld"
  targetNamespace="http://tutorial5.wsrf.dl.ac.uk/helloworld"
  xmlns:tns="http://tutorial5.wsrf.dl.ac.uk/helloworld"
  xmlns:wsrp="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd"
  xmlns:wsrlw="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.wsdl"
  xmlns:wsdlpp="http://www.globus.org/namespaces/2004/10/WSDLPreprocessor"
  xmlns:gtwsdl1="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ServiceGroup-1.2-draft-01.wsdl"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
  xmlns:wsntw="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.wsdl"
  xmlns:wsrpw="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
 <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.wsdl"
  location="../wsrf/faults/WS-BaseFaults.wsdl"/>
 <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.wsdl"
  location="../wsrf/lifetime/WS-ResourceLifetime.wsdl"/>
 <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.wsdl"
  location="../wsrf/properties/WS-ResourceProperties.wsdl"/>
 <import namespace="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.wsdl"
  location="../wsrf/notification/WS-BaseN.wsdl"/>
 <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ServiceGroup-1.2-draft-01.wsdl"
  location="../wsrf/servicegroup/WS-ServiceGroup.wsdl"/>
 <import namespace="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.wsdl"
  location="../wsrf/notification/WS-BaseN.wsdl" />

 <types>
  <schema
    targetNamespace="http://tutorial5.wsrf.dl.ac.uk/helloworld"
    xmlns:tns="http://tutorial5.wsrf.dl.ac.uk/helloworld"
    xmlns="http://www.w3.org/2001/XMLSchema"
```

Destroying the WSRF Resource (Immediate Destruction)

```
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <import namespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"
    schemaLocation="../ws/addressing/WS-Addressing.xsd"/>
  <import  namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ServiceGroup-1.2-draft-01.xsd"
    schemaLocation="../wsrf/servicegroup/WS-ServiceGroup.xsd"/>

   <element name="name" type="xsd:string"/>

   <element name="getNameRPRequest" >
    <complexType/>
   </element>
   <element name="getNameRPResponse" type="xsd:string"/>

   <element name="HelloWorldResourcePropertiesSet">
    <complexType>
     <sequence>
      <element ref="tns:name"/>
     </sequence>
    </complexType>
   </element>

   <element name="echoRequest" type="xsd:string" />
   <element name="echoResponse" type="xsd:string" />
  </schema>
</types>

<message name="EchoRequest">
 <part name="EchoRequest"  element="tns:echoRequest" />
</message>
<message name="EchoResponse">
 <part name="EchoResponse" element="tns:echoResponse" />
</message>

<message name="GetNameRPRequest">
 <part name="GetNameRPRequest" element="tns: getNameRPRequest " />
</message>
<message name="GetNameRPResponse">
 <part name="GetNameRPResponse" element="tns:getNameRPResponse" />
</message>

<portType name="HelloWorldPortType"
      wsrp:ResourceProperties="HelloWorldResourcePropertiesSet">

 <operation name="GetResourceProperty">
  <input name="GetResourcePropertyRequest"
     message="wsrpw:GetResourcePropertyRequest"
     wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/GetResourceProperty"/>
 <output name="GetResourcePropertyResponse"
      message="wsrpw:GetResourcePropertyResponse"
      wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/GetResourcePropertyResponse"/>
 <fault  name="InvalidResourcePropertyQNameFault"
       message="wsrpw:InvalidResourcePropertyQNameFault"/>
```

Destroying the WSRF Resource (Immediate Destruction)

```
    <fault name="ResourceUnknownFault" message="wsrpw:ResourceUnknownFault"/>
   </operation>

  <operation name="Destroy">
     <input message="wsrlw:DestroyRequest"
        wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime/Destroy"/>
     <output message="wsrlw:DestroyResponse"
        wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime/DestroyResponse"/>
     <fault
        message="wsrlw:ResourceNotDestroyedFault" name="ResourceNotDestroyedFault"/>
     <fault message="wsrlw:ResourceUnknownFault"  name="ResourceUnknownFault"/>
   </operation>

   <!--  name in input and output is optional-->
   <operation name="echo">
     <input  name="EchoRequest"  message="tns:EchoRequest" />
     <output name="EchoResponse" message="tns:EchoResponse" />
   </operation>

   <operation name="getNameRP">
     <input  name="GetNameRPRequest"  message="tns:GetNameRPRequest" />
     <output name="GetNameRPResponse" message="tns:GetNameRPResponse" />
   </operation>

  </portType>
</definitions>
```

WSDL for Factory Service

Below is the WSDL file for our Factory Service, which is has only one operation createResource() to create the instance of our main service HelloWorldServie. This WSDL file is in factory5_port_type.wsdl and is exactly the same as factory4_port_type.wsdl, except target name space difference.

```
  <?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloFactoryService5"
   targetNamespace="http://tutorial5.wsrf.dl.ac.uk/helloworld"
   xmlns:tns="http://tutorial5.wsrf.dl.ac.uk/helloworld"
   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>
<xsd:schema targetNamespace="http://tutorial5.wsrf.dl.ac.uk/helloworld"
   xmlns:tns="http://tutorial5.wsrf.dl.ac.uk/helloworld"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">

     <import namespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"
      schemaLocation="../ws/addressing/WS-Addressing.xsd"/>

         <xsd:element name="createResource">
                 <xsd:complexType/>
```

Destroying the WSRF Resource (Immediate Destruction)

```
        </xsd:element>
        <xsd:element name="createResourceResponse">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element ref="wsa:EndpointReference"/>
                </xsd:sequence>
        </xsd:complexType>
        </xsd:element>

</xsd:schema>
</types>

<message name="CreateResourceRequest">
        <part name="request" element="tns:createResource"/>
</message>
<message name="CreateResourceResponse">
        <part name="response" element="tns:createResourceResponse"/>
</message>

<portType name="HelloWorldFactoryPortType">
        <operation name="createResource">
                <input message="tns:CreateResourceRequest"/>
                <output message="tns:CreateResourceResponse"/>
        </operation>
</portType>

</definitions>
```

Modified Implementation of Web Service:

This implementation is using the similar interface which we used in last example and only change is in the NS variable which is changed from http://tutorial4.wsrf.dl.ac.uk/helloworld to http://tutorial5.wsrf.dl.ac.uk/helloworld .

```
        public static final String NS = "http://tutorial5.wsrf.dl.ac.uk/helloworld";
```

```
package uk.ac.dl.ws.service;

import javax.xml.namespace.QName;

public interface HelloQNames {
   public static final String NS = "http://tutorial5.wsrf.dl.ac.uk/helloworld";
   public static final QName RP_NAME = new QName(NS, "name");
   public static final QName RESOURCE_PROPERTIES = new QName(NS, "HelloWorldResourcePropertiesSet");
}
```

String NS is String value of our target namespace which is http://tutorial5.wsrf.dl.ac.uk/helloworld for tutorial 5. RP_NAME and RESOURCE_PROPERTIES are qualified name of our variable declared in our WSDL file. Only package we have to import is "*javax.xml.namespace.QName*" as this interface is utility interface and has nothing to do with WSRF implementation.

There is no change in any of the implementation classes and they are exactly the same as last tutorial, except the changes in the name, lack of blue font in the table below indicates the extensive similarities in classes.

Destroying the WSRF Resource (Immediate Destruction)

Tutorial 5

```
package uk.ac.dl.ws.service;

import org.globus.wsrf.Resource;
import org.globus.wsrf.ResourceProperties;
import org.globus.wsrf.ResourceProperty;
import org.globus.wsrf.ResourcePropertySet;
import org.globus.wsrf.ResourceIdentifier;
import org.globus.wsrf.impl.SimpleResourceProperty;
import org.globus.wsrf.impl.SimpleResourcePropertySet;

public class HelloWorldResourceFactory5
    implements Resource, ResourceProperties, ResourceIdentifier {

  /** the identifier of Resource */
  private Object id;

  /** Stores the ResourceProperties of HelloWorld Service */
  private ResourcePropertySet propSet;

  /* Variable for Resource property */
  private String name;

  /** Resource property "name". */
  private ResourceProperty nameRP;

  /** initializes the HelloWorld. */
  public void initialize() throws Exception {
    // choose an ID
    this.id = new Integer(hashCode());

    // create the resource property set
    this.propSet = new SimpleResourcePropertySet(
        HelloQNames.RESOURCE_PROPERTIES);
    // create resource properties
    this.nameRP = new
        SimpleResourceProperty(HelloQNames.RP_NAME);

    this.propSet.add(this.nameRP);
    setName("Asif Akram from Tutorial 5");
    this.nameRP.add(name);
  }
  public ResourcePropertySet getResourcePropertySet() {
    return propSet;
  }
  public void setNameRP(String nameValue) {
    setName(nameValue);
    this.nameRP.set(0, name);
  }
  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
  public Object getID() {
    return id;
  }
}
```

Tutorial 4

```
package uk.ac.dl.ws.service;

import org.globus.wsrf.Resource;
import org.globus.wsrf.ResourceProperties;
import org.globus.wsrf.ResourceProperty;
import org.globus.wsrf.ResourcePropertySet;
import org.globus.wsrf.ResourceIdentifier;
import org.globus.wsrf.impl.SimpleResourceProperty;
import org.globus.wsrf.impl.SimpleResourcePropertySet;

public class HelloWorldResourceFactory
    implements Resource, ResourceProperties, ResourceIdentifier {

  /** the identifier of this Resource */
  private Object id;

  /** Stores the ResourceProperties */
  private ResourcePropertySet propSet;

  /* Variable for Resource property */
  private String name;

  /** Resource property "name". */
  private ResourceProperty nameRP;

  /** initializes the HelloWorld. */
  public void initialize() throws Exception {
    // choose an ID
    this.id = new Integer(hashCode());

    // create the resource property set
    this.propSet = new SimpleResourcePropertySet(
        HelloQNames.RESOURCE_PROPERTIES);
    // create resource properties
    this.nameRP = new
        SimpleResourceProperty(HelloQNames.RP_NAME);

    this.propSet.add(this.nameRP);
    setName("Asif Akram from Tutorial 4");
    this.nameRP.add(name);
  }
  public ResourcePropertySet getResourcePropertySet() {
    return propSet;
  }
  public void setNameRP(String nameValue) {
    setName(nameValue);
    this.nameRP.set(0, name);
  }
  public String getName() {
    return name;
  }
  public void setName(String name) {
    this.name = name;
  }
  public Object getID() {
    return id;
  }
}
```

package uk.ac.dl.ws.service;

package uk.ac.dl.ws.service;

Destroying the WSRF Resource (Immediate Destruction)

```
import org.globus.wsrf.ResourceKey;
import org.globus.wsrf.impl.ResourceHomeImpl;
import org.globus.wsrf.impl.SimpleResourceKey;

public class HelloWorldResourceHomeFactory5 extends
ResourceHomeImpl{

 public ResourceKey create() {
  try {
   HelloWorldResourceFactory5 hello =
      (HelloWorldResourceFactory5) createNewInstance();
   hello.initialize();
   ResourceKey key = new
       SimpleResourceKey(keyTypeName, hello.getID());
   this.add(key, hello);
   return key;
  }
  catch (Exception e) {
   System.out.println("Exception when creating HelloWorld: ");
   e.printStackTrace();
   return null;
  }

 }

}
```

```
import org.globus.wsrf.ResourceKey;
import org.globus.wsrf.impl.ResourceHomeImpl;
import org.globus.wsrf.impl.SimpleResourceKey;

public class HelloWorldResourceHomeFactory extends
ResourceHomeImpl{

 public ResourceKey create() {
  try {
   HelloWorldResourceFactory hello =
      (HelloWorldResourceFactory) createNewInstance();
   hello.initialize();
   ResourceKey key = new
       SimpleResourceKey(keyTypeName, hello.getID());
   this.add(key, hello);
   return key;
  }
  catch (Exception e) {
   System.out.println("Exception when creating HelloWorld: ");
   e.printStackTrace();
   return null;
  }

 }

}
```

```
package uk.ac.dl.ws.service;

import java.rmi.RemoteException;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.globus.wsrf.ResourceContext;
import org.globus.wsrf.ResourceKey;
import org.globus.wsrf.utils.AddressingUtils;
import uk.ac.dl.wsrf.tutorial5.helloworld.*;

public class HelloWorldService5 {

 public HelloWorldService5() throws RemoteException { }

 public String echo(String name) throws RemoteException {
  HelloWorldResourceFactory5 helloWorldResource = getResource();
  helloWorldResource.setNameRP(name);
  return "Hello " + name + " !";
 }
 public String getNameRP(GetNameRPRequest params) throws
                                                RemoteException {
  HelloWorldResourceFactory5 helloWorldResource = getResource();
  return helloWorldResource.getName();
  //return "This method is working Tutorial 3";
 }
 public HelloWorldResourceFactory5 getResource() throws
RemoteException {
  Object resource = null;
  try {
   resource =
       ResourceContext.getResourceContext().getResource();
  } catch (Exception e) {
   throw new RemoteException("", e);
  }
  HelloWorldResourceFactory5 helloWorldResource =
           (HelloWorldResourceFactory5) resource;
```

```
package uk.ac.dl.ws.service;

import java.rmi.RemoteException;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.globus.wsrf.ResourceContext;
import org.globus.wsrf.ResourceKey;
import org.globus.wsrf.utils.AddressingUtils;
import uk.ac.dl.wsrf.tutorial4.helloworld.*;

public class HelloWorldService {

 public HelloWorldService() throws RemoteException { }

 public String echo(String name) throws RemoteException {
  HelloWorldResourceFactory helloWorldResource = getResource();
  helloWorldResource.setNameRP(name);
  return "Hello " + name + " !";
 }
 public String getNameRP(GetNameRPRequest params) throws
                                                RemoteException {
  HelloWorldResourceFactory helloWorldResource = getResource();
  return helloWorldResource.getName();
  //return "This method is working Tutorial 3";
 }
 public HelloWorldResourceFactory getResource() throws
RemoteException {
  Object resource = null;
  try {
   resource =
       ResourceContext.getResourceContext().getResource();
  } catch (Exception e) {
   throw new RemoteException("", e);
  }
  HelloWorldResourceFactory helloWorldResource =
           (HelloWorldResourceFactory) resource;
```

Destroying the WSRF Resource (Immediate Destruction)

```
    return helloWorldResource;
  }
}
```

```
    return helloWorldResource;
  }
}
```

```
package uk.ac.dl.ws.service;

import java.rmi.RemoteException;
import java.net.URL;

import org.globus.wsrf.ResourceContext;
import org.globus.wsrf.ResourceKey;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.apache.axis.MessageContext;
import org.globus.wsrf.utils.AddressingUtils;
import org.globus.wsrf.container.ServiceHost;
import uk.ac.dl.wsrf.tutorial5.helloworld.*;

public class HelloWorldFactoryService5{

  /* Implementation of createResource Operation */
  public CreateResourceResponse createResource(CreateResource
                        request) throws  RemoteException {
    ResourceContext ctx = null;
    HelloWorldResourceHomeFactory5 home = null;
    ResourceKey key = null;

    /* First, we create a new HelloWorldResourceFactory through
            the HelloWorldResourceHomeFactory */
    try {
      ctx = ResourceContext.getResourceContext();
      home = (HelloWorldResourceHomeFactory5)
          ctx.getResourceHome();
      key = home.create();
    }
    catch (Exception e) {
      throw new RemoteException("", e);
    }
    EndpointReferenceType epr = null;

    /* We construct the instance's endpoint reference. The instance's
service     * path can be found in the WSDD file as a parameter. */
    try {
      URL baseURL = ServiceHost.getBaseURL();
      String instanceService = (String) MessageContext
          .getCurrentContext().getService().getOption("instance");
      String instanceURI = baseURL.toString() + instanceService;
      // The endpoint reference includes instance's URI and resource key
      epr = AddressingUtils.createEndpointReference(instanceURI, key);
    }
    catch (Exception e) {
      throw new RemoteException("", e);
    }

    /* Finally, return endpoint reference in a CreateResourceResponse */
    CreateResourceResponse response = new
                          CreateResourceResponse();
    response.setEndpointReference(epr);
    return response;
  }
}
```

```
package uk.ac.dl.ws.service;

import java.rmi.RemoteException;
import java.net.URL;

import org.globus.wsrf.ResourceContext;
import org.globus.wsrf.ResourceKey;
import org.apache.axis.message.addressing.EndpointReferenceType;
import org.apache.axis.MessageContext;
import org.globus.wsrf.utils.AddressingUtils;
import org.globus.wsrf.container.ServiceHost;
import uk.ac.dl.wsrf.tutorial4.helloworld.*;

public class HelloWorldFactoryService{

  /* Implementation of createResource Operation */
  public CreateResourceResponse createResource(CreateResource
                        request) throws RemoteException {
    ResourceContext ctx = null;
    HelloWorldResourceHomeFactory home = null;
    ResourceKey key = null;

    /* First, we create a new HelloWorldResourceFactory through
            the HelloWorldResourceHomeFactory  */
    try {
      ctx = ResourceContext.getResourceContext();
      home = (HelloWorldResourceHomeFactory)
          ctx.getResourceHome();
      key = home.create();
    }
    catch (Exception e) {
      throw new RemoteException("", e);
    }
    EndpointReferenceType epr = null;

    /* We construct the instance's endpoint reference. The instance's
service  path can be found in the WSDD file as a parameter. */
    try {
      URL baseURL = ServiceHost.getBaseURL();
      String instanceService = (String) MessageContext
          .getCurrentContext().getService().getOption("instance");
      String instanceURI = baseURL.toString() + instanceService;
      // The endpoint reference includes instance's URI and resource key
      epr = AddressingUtils.createEndpointReference(instanceURI, key);
    }
    catch (Exception e) {
      throw new RemoteException("", e);
    }

    /* Finally, return endpoint reference in createResourceResponse */
    CreateResourceResponse response = new
          CreateResourceResponse();
    response.setEndpointReference(epr);
    return response;
  }
}
```

Modified Deployment Descriptor:

Destroying the WSRF Resource (Immediate Destruction)

Deployment Descriptor is also nearly the same as last tutorial, nothing to change except name of Services, WSDL files and name of classes implementing Services. The only significance change is adding "DestroyProvider" as one of the value for "providers".

```
<deployment name="defaultServerConfig" xmlns="http://xml.apache.org/axis/wsdd/"
xmlns:aggr="http://mds.globus.org/aggregator/types"
xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

 <service name="HelloWorldService5" provider="Handler" use="literal" style="document">
  <parameter name="providers" value="GetRPProvider DestroyProvider"/>
  <parameter name="handlerClass" value="org.globus.axis.providers.RPCProvider"/>
  <parameter name="scope" value="Application"/>
  <parameter name="allowedMethods" value="*"/>
  <parameter name="activateOnStartup" value="true"/>
  <parameter name="className" value="uk.ac.dl.ws.service.HelloWorldService5"/>
  <wsdlFile>share/schema/tutorial/HelloWorld5_service.wsdl</wsdlFile>
 </service>

 <! -- Factory service -->
  <service name="HelloFactoryService5" provider="Handler" use="literal" style="document">
      <parameter name="handlerClass" value="org.globus.axis.providers.RPCProvider"/>
     <parameter name="scope" value="Application"/>
     <parameter name="allowedMethods" value="*"/>
     <parameter name="instance" value="HelloWorldService5"/>
     <parameter name="className" value="uk.ac.dl.ws.service.HelloWorldFactoryService5"/>
     <wsdlFile>share/schema/tutorial/FactoryService5_service.wsdl</wsdlFile>
  </service>

</deployment>
```

Tutorial 5	Tutorial 4

`<deployment name="defaultServerConfig"`
`xmlns="http://xml.apache.org/axis/wsdd/"`
`xmlns:aggr="http://mds.globus.org/aggregator/types"`
`xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"`
`xmlns:xsd="http://www.w3.org/2001/XMLSchema">`
`<service name="HelloWorldService5" provider="Handler"`
`use="literal" style="document">`
`<parameter name="providers" value="GetRPProvider`
`DestroyProvider"/>`
`<parameter name="handlerClass"`
`value="org.globus.axis.providers.RPCProvider"/>`
`<parameter name="scope" value="Application"/>`
`<parameter name="allowedMethods" value="*"/>`
`<parameter name="activateOnStartup" value="true"/>`
`<parameter name="className"`
`value="uk.ac.dl.ws.service.HelloWorldService5"/>`
`<wsdlFile>`
`share/schema/tutorial/HelloWorld5_service.wsdl`
`</wsdlFile>`
`</service>`
 	
`<!-- Factory service -->`
`<service name="HelloFactoryService5" provider="Handler"`
`use="literal" style="document">`
	`<deployment name="defaultServerConfig"`
	`xmlns="http://xml.apache.org/axis/wsdd/"`
	`xmlns:aggr="http://mds.globus.org/aggregator/types"`
	`xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"`
	`xmlns:xsd="http://www.w3.org/2001/XMLSchema">`
	`<service name="HelloWorldService4" provider="Handler"`
	`use="literal" style="document">`
	`<parameter name="providers" value="GetRPProvider"/>`
	
	`<parameter name="handlerClass"`
	`value="org.globus.axis.providers.RPCProvider"/>`
	`<parameter name="scope" value="Application"/>`
	`<parameter name="allowedMethods" value="*"/>`
	`<parameter name="activateOnStartup" value="true"/>`
	`<parameter name="className"`
	`value="uk.ac.dl.ws.service.HelloWorldService"/>`
	`<wsdlFile>`
	`share/schema/tutorial/HelloWorld4_service.wsdl`
	`</wsdlFile>`
	`</service>`
	
	`<!-- Factory service -->`
	`<service name="HelloFactoryService" provider="Handler"`
	`use="literal" style="document">`

Destroying the WSRF Resource (Immediate Destruction)

```
    <parameter name="handlerClass"
          value="org.globus.axis.providers.RPCProvider"/>
    <parameter name="scope" value="Application"/>
    <parameter name="allowedMethods" value="*"/>
    <parameter name="instance" value="HelloWorldService5"/>
    <parameter name="className"
       value="uk.ac.dl.ws.service.HelloWorldFactoryService5"/>
    <wsdlFile>
          share/schema/tutorial/FactoryService5_service.wsdl
    </wsdlFile>
   </service>
</deployment>
```

```
    <parameter name="handlerClass"
          value="org.globus.axis.providers.RPCProvider"/>
    <parameter name="scope" value="Application"/>
    <parameter name="allowedMethods" value="*"/>
    <parameter name="instance" value="HelloWorldService4"/>
    <parameter name="className"
          value="uk.ac.dl.ws.service.HelloWorldFactoryService"/>
    <wsdlFile>
          share/schema/tutorial/FactoryService_service.wsdl
    </wsdlFile>
  </service>
</deployment>
```

Modified Deployment "jndi-config":

The "deploy-jndi-config.xml" Deployment Descriptor is also nearly the same as last tutorial, nothing to change except target name space and Service name.

```
<jndiConfig xmlns="http://wsrf.globus.org/jndi/config">

  <service name="HelloWorldService5">
    <resource
      name="home" type="uk.ac.dl.ws.service.HelloWorldResourceHomeFactory5">
      <resourceParams>
        <parameter>
          <name>factory</name>
          <value>org.globus.wsrf.jndi.BeanFactory</value>
        </parameter>
        <parameter>
          <name>resourceClass</name>
          <value>uk.ac.dl.ws.service.HelloWorldResourceFactory5</value>
        </parameter>
        <parameter>
          <name>resourceKeyName</name>
          <value>{http://tutorial5.wsrf.dl.ac.uk/helloworld}NameKeyFactory5</value>
        </parameter>
        <parameter>
          <name>resourceKeyType</name>
          <value>java.lang.Integer</value>
        </parameter>
      </resourceParams>
    </resource>
  </service>

  <!-- Factory service -->
  <service name="HelloFactoryService5">
    <resourceLink name="home" target="java:comp/env/services/HelloWorldService5/home"/>
  </service>

</jndiConfig>
```

Tutorial 5	Tutorial 4
<jndiConfig xmlns="http://wsrf.globus.org/jndi/config">	<jndiConfig xmlns="http://wsrf.globus.org/jndi/config">

Destroying the WSRF Resource (Immediate Destruction)

```
<service name="HelloWorldService5">
  <resource name="home"
   type="uk.ac.dl.ws.service.HelloWorldResourceHomeFactory5">
    <resourceParams>
      <parameter>
        <name>factory</name>
        <value>org.globus.wsrf.jndi.BeanFactory</value>
      </parameter>
      <parameter>
        <name>resourceClass</name>
        <value>
          uk.ac.dl.ws.service.HelloWorldResourceFactory5
        </value>
      </parameter>
      <parameter>
        <name>resourceKeyName</name>
        <value>
        {http://tutorial5.wsrf.dl.ac.uk/helloworld}NameKeyFactory5
       </value>
      </parameter>
      <parameter>
        <name>resourceKeyType</name>
        <value>java.lang.Integer</value>
      </parameter>
    </resourceParams>
  </resource>
</service>

<!-- Factory service -->
<service name="HelloFactoryService5">
  <resourceLink name="home"
target="java:comp/env/services/HelloWorldService5/home"/>
</service>
</jndiConfig>
```

```
<service name="HelloWorldService4">
  <resource name="home"
    type="uk.ac.dl.ws.service.HelloWorldResourceHomeFactory">
    <resourceParams>
      <parameter>
        <name>factory</name>
        <value>org.globus.wsrf.jndi.BeanFactory</value>
      </parameter>
      <parameter>
        <name>resourceClass</name>
        <value>
          uk.ac.dl.ws.service.HelloWorldResourceFactory
        </value>
      </parameter>
      <parameter>
        <name>resourceKeyName</name>
        <value>
        {http://tutorial4.wsrf.dl.ac.uk/helloworld}NameKeyFactory
        </value>
      </parameter>
      <parameter>
        <name>resourceKeyType</name>
        <value>java.lang.Integer</value>
      </parameter>
    </resourceParams>
  </resource>
</service>

<!-- Factory service -->
<service name="HelloFactoryService">
    <resourceLink name="home"
      target="java:comp/env/services/HelloWorldService4/home"/>
  </service>
</jndiConfig>
```

Modified Implementation of our Client:

Last thing to do is to write Client to test our WSRF Web Service. Below is the complete code of Client5.java followed by discussing important parts, notice that Client5.java is in package "uk.ac.dl.ws.service.*client*" and our service HelloWorld.java is in package "*package* uk.ac.dl.ws.service". This is not the requirement of WSRF or Globus implementation of WSRF, in fact it is restriction due to the Ant "build.xml" file which will be used later to generate stubs, compile service and deploy to the container. You can change "build.xml" to adjust changes made in your package structure. We have to put all java classes and additional files like WSDL and "deploy-jndi-config.xml" in specific directories as required by Ant "build.xml". Ant "build.xml" is so handy and useful that following few restrictions is still worthy, as it hides all dirty work of setting class path, copying files from different locations and above all making all imported files in our WSDL available to our WSDL file.

```
package uk.ac.dl.ws.service.client;

import org.oasis.wsrf.properties.WSResourcePropertiesServiceAddressingLocator;
import org.oasis.wsrf.properties.GetResourceProperty;
import org.oasis.wsrf.properties.GetResourcePropertyResponse;
import org.apache.axis.message.addressing.EndpointReferenceType;

import org.apache.commons.cli.ParseException;
import org.apache.commons.cli.CommandLine;

import org.globus.wsrf.client.BaseClient;
import org.globus.wsrf.encoding.ObjectSerializer;
import org.globus.wsrf.encoding.SerializationException;
```

Destroying the WSRF Resource (Immediate Destruction)

```
import org.globus.wsrf.utils.AnyHelper;
import org.globus.wsrf.utils.FaultHelper;

import org.oasis.wsrf.lifetime.Destroy;

import java.io.FileWriter;
import java.io.IOException;

import java.util.List;
import java.util.Random;

import javax.xml.rpc.Stub;
import uk.ac.dl.ws.service.HelloQNames;
import uk.ac.dl.wsrf.tutorial5.helloworld.service.*;
import uk.ac.dl.wsrf.tutorial5.helloworld.*;

public class Client5 extends BaseClient {

 final static HelloWorld5ServiceAddressingLocator locator = new HelloWorld5ServiceAddressingLocator();

 final static HelloFactoryService5ServiceAddressingLocator factoryLocator =
    new HelloFactoryService5ServiceAddressingLocator();

 public static void main(String[] args) {
  Client5 client = new Client5();

  // first, parse the commandline
  try {
    CommandLine line = client.parse(args);
  }
  catch (ParseException e) {
    System.err.println("Parsing failed: " + e.getMessage());
    System.exit(1);
  }
  catch (Exception e) {
    System.err.println("Error: " + e.getMessage());
    System.exit(1);
  }

  try {
    EndpointReferenceType instanceEPR;
    HelloWorldPortType port;
    HelloWorldFactoryPortType factoryPort = factoryLocator.getHelloWorldFactoryPortTypePort(client.getEPR());
    CreateResourceResponse createResponse = factoryPort.createResource(new CreateResource());

    instanceEPR = createResponse.getEndpointReference();
    System.out.println("new resource created...");

    instanceEPR.getAddress().setPort(8082);

    System.out.println("Before Creating instance.");
    // Get instance PortType
    port = locator.getHelloWorldPortTypePort(instanceEPR);
    System.out.println("Instance Created.");


    GetResourcePropertyResponse response = port.getResourceProperty(HelloQNames.RP_NAME);
    System.out.println(AnyHelper.toSingleString(response));

    String result = port.echo("Rob Allan First Time Tutorial-5");
    response =port.getResourceProperty(HelloQNames.RP_NAME);
    System.out.println(AnyHelper.toSingleString(response));
```

Destroying the WSRF Resource (Immediate Destruction)

```
    result = port.echo("Rob Allan Second Time Tutorial-5");
    response =port.getResourceProperty(HelloQNames.RP_NAME);
    System.out.println(AnyHelper.toSingleString(response));

    System.out.println(port.getNameRP(new GetNameRPRequest()));
    port.destroy(new Destroy());
    System.out.println(port.getNameRP(new GetNameRPRequest()));
   }
   catch (Exception e) {
    if (client.isDebugMode()) {
     FaultHelper.printStackTrace(e);
    }
    else {
     System.err.println("Error: " + FaultHelper.getMessage(e));
    }
   }
  }
}
```

Implementation of Client once again starts with importing couple of Globus dependent java classes along with importing few classes which are not yet available and will created by Ant build file. Two of them are to locate our Web Service *"HelloWorldService5AddressingLocator"and* "HelloFactoryService5ServiceAddressingLocator" which are in the package *"uk.ac.dl.wsrf.tutorial5.helloworld.service"* and the *"HelloWorldPortType" and* "HelloWorldFactoryPortType" which are the interfaces, "HelloWorldFactoryPortType" had the createRoesource() and *"HelloWorldPortType"* has remaining two methods available in our main Web Service. *"HelloWorldPortType"and* "HelloWorldFactoryPortType" are available in the package *"uk.ac.dl.wsrf.tutorial5.helloworld".* Our client implementation is using HelloFactoryService5ServiceAddressingLocator to create custom stub implementing HelloWorldFactoryPortType, this stub has createResource() method which returns the EndpointReferenceType for our HelloWorld5Service.

```
HelloWorldFactoryPortType factoryPort =
        factoryLocator.getHelloWorldFactoryPortTypePort(client.getEPR());
CreateResourceResponse createResponse = factoryPort .createResource(new CreateResource());
EndpointReferenceType  instanceEPR = createResponse.getEndpointReference();
```

HelloWorldServiceAddressingLocator is used to create client stub HelloWorldPortType using the EndpointReferenceType returned by the createResource() of HelloWorldFactoryPortType, through this stub we have access to two operations exposed by WSDL as shown in following code fragment;

```
HelloWorld5ServiceAddressingLocator locator = new HelloWorld5ServiceAddressingLocator();
HelloWorldPortType port = locator.getHelloWorldPortTypePort(instanceEPR);
```

Client 5 calls the Destroy() operation of HelloWorldService5 after calling the echo() twice. This will destroy that specific resource, whose EndpointReferenceType was used to create the instance of *HelloWorldPortType.*

```
port.destroy(new Destroy());
```

After destroying the resource Client5 tries to get the value of resource, which will give the error.
Client4 and Client5 are very similar and doing exactly the same job, except Client5 destroys the resource and try t o access it again, which gives the error. Client5 goes through following steps:

1. Create instance of HelloFactoryService5ServiceAddressingLocator.
2. Get instance of *HelloWorldFactoryPortType* from HelloFactoryService5ServiceAddressingLocator.
3. Call createResource() operation of *HelloWorldFactoryPortType to get* EndpointReferenceType of HelloWorldService5.
4. Create instance of *HelloWorld5ServiceAddressingLocator*.

Destroying the WSRF Resource (Immediate Destruction)

5. Get instance of *HelloWorldPortType* from *HelloWorld5ServiceAddressingLocator* by passing the EndpointReferenceType obtained in step 3.
6. Use *HelloWorldPortType* to call business methods in the HelloWorldService5.
7. Call Destroy operation for HelloWorldService5 through instance of *HelloWorldPortType*.
8. Retrieve the value of "nameRP", which will give the error.

Modified "post-deploy":

Now everything is available and we have to build the WSRF Web Service. For build purposes we will use build tool "ANT" and the build.xml provided by the Globus. To make life easy to test web service we need last file "post-deploy.xml" which will create script to run the client, without setting class path, without worrying about the generated stubs and location of compiled stubs. This xml file is generated when everything goes well, below are the contents of file "post-deploy.xml".

```
<project default="all" basedir=".">
   <property environment="env"/>
   <property file="build.properties"/>
   <property file="${user.home}/build.properties"/>
   <property name="env.GLOBUS_LOCATION" value="."/>
   <property name="deploy.dir" location="${env.GLOBUS_LOCATION}"/>
   <property name="abs.deploy.dir" location="${deploy.dir}"/>
   <property name="build.launcher"
     location="${abs.deploy.dir}/share/globus_wsrf_common/build-launcher.xml"/>

   <target name="setup">
    <ant antfile="${build.launcher}"
      target="generateLauncher">
      <property name="launcher-name" value="helloworld5"/>
      <property name="class.name"  value="uk.ac.dl.ws.service.client.Client5"/>
    </ant>
  </target>
</project>
```

This file is quite simple, and most of time remains same for most of clients. There are two adjustments to be made for each client and location of those adjustments is shown in bold. Name of the generated script: ***<property name="launcher-name" value="helloworld5"/>*** and location and name of java client of WSRF Web Service. In our case client is "*Client5.java*" in package "***uk.ac.dl.ws.service.client":***

```
<property name="class.name"    value="uk.ac.dl.ws.service.client.Client5"/>
```

Tutorial 5: *"post-deploy.xml"*	Tutorial 4: *"post-deploy.xml"*
<project default="all" basedir="."> <property environment="env"/> <property file="build.properties"/> <property file="${user.home}/build.properties"/> <property name="env.GLOBUS_LOCATION" value="."/> <property name="deploy.dir" location="${env.GLOBUS_LOCATION}"/> <property name="abs.deploy.dir" location="${deploy.dir}"/> <property name="build.launcher" location="${abs.deploy.dir}/share/globus_wsrf_common/build-launcher.xml"/> <target name="setup">	<project default="all" basedir="."> <property environment="env"/> <property file="build.properties"/> <property file="${user.home}/build.properties"/> <property name="env.GLOBUS_LOCATION" value="."/> <property name="deploy.dir" location="${env.GLOBUS_LOCATION}"/> <property name="abs.deploy.dir" location="${deploy.dir}"/> <property name="build.launcher" location="${abs.deploy.dir}/share/globus_wsrf_common/build-launcher.xml"/> <target name="setup">

Destroying the WSRF Resource (Immediate Destruction)

```
    <ant antfile="${build.launcher}"
        target="generateLauncher">
      <property name="launcher-name" value="helloworld5"/>
      <property name="class.name"
          value="uk.ac.dl.ws.service.client.Client5"/>
    </ant>
  </target>
</project>
```

```
    <ant antfile="${build.launcher}"
        target="generateLauncher">
      <property name="launcher-name" value="helloworld4"/>
      <property name="class.name"
          value="uk.ac.dl.ws.service.client.Client4"/>
    </ant>
  </target>
</project>
```

Modified Ant Script:

Now we have to use Ant build file, but before that we have to change the build file which are very minimum changes i.e. name changes (which we are doing in all tutorials). The changes are only about changing the name of WSDL files generated by the Globus, in last tutorials we had added another Service thus we have two Services, HelloWorldService5 and HelloWorldFactoryService5. These changes are the last step required to finish the whole implementation.

Once again I have copied the complete build file and highlighted the changes related to Tutorial 5 in Bold Green Font:

```
<?xml version="1.0" encoding="UTF-8"?>
<project basedir="." default="all" name="globus_tutorial5_ helloworld ">
  <description>
    WSRF MDS Aggregator Implementation
  </description>

  <property environment="env"/>

  <property file="build.propert_properties"/>
  <property file="${user.home}/build.properties"/>

  <property location="../install" name="env.GLOBUS_LOCATION"/>
  <property location="${env.GLOBUS_LOCATION}" name="deploy.dir"/>
  <property name="base.name" value="tutorial5_helloworld"/>
  <property name="package.namç" value="globus_${base.name}"/>
  <property name="jar.name" value="${package.name}.jar"/>
  <property name="gar.name" value="${package.name}.gar"/>
  <property location="build" name="build.dir"/>
  <property location="build/class_'14" name="build.dest"/>
  <property location="build/lib" name="build.lib.xir"/>
  <property location="build/stubs" name="stubs.dir"/>
  <property location="build/stubs/_rc" name="stubs.src"/>
  <property location="build/stubs/classes" name="stubs.dest"/>
  <property name="stubs.jar.name" value="${package.name}_stubs.jar"/>
  <property location="${deploy.dir}/share/globus_wsrf_common/build-packages.xml"
        name="build.packages"/>
  <property location="${deploy.dir}/share/globus_wsrf_tools/build-stubs.xml" name="build.stubs"/>
  <property name="java.debug" value="on"/>

  <property name="test-reports.dir" value="test-reports"/>
  <property name="junit.haltonfailure" value="true"/>
```

Destroying the WSRF Resource (Immediate Destruction)

```
<property location="${deploy.dir}/share/schema" name="schema.src"/>
<property location="${build.dir}/schema" name="schema.dest"/>

<property name="garjars.id" value="garjars"/>
<fileset dir="${build.lib.d_u114 ?}" id="garjars"/>

<property name="garschema.id" value="garschema"/>
<fileset dir="${schema.dest}" id="garschema" includes="tutorial/*"/>

<property name="garetc.id" value="garetc"/>
<fileset dir="etc" id="garetc"/>

<target name="init">
   <mkdir dir="${build.dir}"/>
   <mkdir dir="${build.dest}"/>
   <mkdir dir="${build.lib.d_u114 ?}"/>

   <mkdir dir="${stubs.dir}"/>
   <mkdir dir="${stubs.src}"/>
   <mkdir dir="${stubs.dest}"/>

   <mkdir dir="${schema.dest}"/>

   <copy toDir="${schema.dest}">
     <fileset dir="${schema.src}" casesensitive="yes">
       <include name="wsrf/**/*"/>
       <include name="ws/**/*"/>
     </fileset>
   </copy>
   <copy toDir="${schema.dest}">
     <fileset dir="schema/" casesensitive="yes">
       <include name="tutorial/*"/>
     </fileset>
   </copy>

   <available file="${stubs.dest}/org.globus.wsrf.mds.aggregator" property="stubs.present" type="dir"/>
</target>

<target name="bindings" depends="init">
<ant antfile="${build.stub_}" target="generateBindin,">
      <property name="source.binding.dir"
        value="${schema.dest}/tutorial"/>
      <property name="target.binding.dir"
        value="${schema.dest}/tutorial"/>
      <property name="binding.root" value="HelloWorld5"/>
      <property name="porttype.wsdl" value="helloworld5_port_type.wsdl"/>
   </ant>
</target>

<target name="bindingsFactory" depends="bindings">
<ant antfile="${build.stub_}" target="generateBindin(">
  <property name="source.bin€u105 ?ng.dir"
```

```
      value="${schema.dest}/tutorial"/>
    <property name="target.binxing.dir"
      value="${schema.dest}/tutorial"/>
    <property name="binding.root" value="FactoryService5"/>
    <property name="porttype.wsdl" value="factory5_port_type.wsdl"/>
  </ant>
 </target>

 <target name="stubs" unless="stubs.present" depends="bindingsFactory">
    <ant antfile="${build.stub_}" target="generateStub_">
      <property location="${schema.dest}/tutorial" name="source.stubs.dir"/>
      <property name="wsdl.file" value="HelloWorld5__service.wsdl"/>
      <property location="${stubs.src}" name="target.stubs.dir"/>
    </ant>
 </target>

 <target name="stubsFactory" unless="stubs.present" depends="stubs">
    <ant antfile="${build.stub_}" target="generateStub_">
      <property location="${schema.dest}/tutorial" name="source.stubs.dir"/>
      <property name="wsdl.file" value="FactoryService5_service.wsdl"/>
      <property location="${stubs.src}" name="target.stubs.dir"/>
    </ant>
 </target>

 <target depends="stubsFactory" name="compileStubs">
    <delete dir="${stubs.src}/org/apache"/>
    <javac debug="${java.debug}" destdir="${stubs.dest}" srcdir="${stubs.src}">
      <include name="**/*.java"/>
      <classpath>
        <fileset dir="${deploy.dir}/lib">
          <include name="*.jar"/>
        </fileset>
      </classpath>
    </javac>
 </target>

 <target depends="compileStubs" name="compile">
    <javac debug="${java.debug}" destdir="${build.dest}" srcdir="src">
      <include name="**/*.java"/>
      <classpath>
        <pathelement location="${stubs.dest}"/>
        <fileset dir="${deploy.dir}/lib">
          <include name="*.jar"/>
        </fileset>
      </classpath>
    </javac>
 </target>

 <target depends="compileStubs" name="jarStubs">
    <jar basedir="${stubs.dest}" destfile="${build.lib.d_u114 ?}/${stubs.jar.name}"/>
 </target>
```

Destroying the WSRF Resource (Immediate Destruction)

```
<target depends="compile" name="jar">
  <jar basedir="${build.dest}" destfile="${build.lib.d_u114 ?}/${jar.name}"/>
</target>

<target depends="jar, jarStub_" name="dist">
  <ant antfile="${build.packages}" target="makeGar">
    <property name="gar.name" value="${build.lib.d_u114 ?}/${gar.name}"/>
    <reference refid="${garjars.id}"/>
    <reference refid="${garschema.id}"/>
    <reference refid="${garetc.i€u125 ?"/>
  </ant>
</target>

<target name="clean">
  <delete dir="tmp"/>
  <delete dir="${build.dir}"/>
  <delete file="${gar.name}"/>
  <delete dir="${test-reports.dir}"/>
</target>

<target depends="dist" name="deploy">
  <ant antfile="${build.packages}" target="deployGar">
    <property name="gar.name" value="${build.lib.d_u114 ?}/${gar.name}"/>
    <property name="gar.id" value="${package.name}"/>
    <!-- <property name="noSchema" value="true"/> -->
  </ant>
</target>

<target name="undeploy">
  <ant antfile="${build.packages}" target="undeployGar">
    <property name="gar.id" value="${package.name}"/>
  </ant>
</target>

<target depends="deploy" name="all"/>

<target depends="compile" name="test">
  <mkdir dir="${test-reports.dir}"/>
  <junit fork="yes" haltonfailure="${junit.haltonfailure}" printsummary="yes" timeout="600000">
    <classpath>
      <pathelement location="${build.dest}"/>
      <pathelement location="${deploy.dir}"/>
      <pathelement location="${deploy.dir}/etc/wsrf-bundles.properties"/>
      <fileset dir="${deploy.dir}/lib">
        <include name="*.jar"/>
      </fileset>
    </classpath>
    <formatter type="xml"/>
    <batchtest todir="${test-reports.dir}">
      <fileset dir="${build.dest}">
        <include name="**/*Test.class"/>
```

```
        </fileset>
      </batchtest>
    </junit>
  </target>

</project>
```

There are only 8 minor changes in the Ant Script from Tutorial 4; those changes are highlighted in the Bold Green Font. Changes in the script are more related to changing the name of WSDL file created by us for Tutorial 5 i.e. "*helloworld5_port_type.wsdl*" and "*factory5_port_type.wsdl*", and the name of final WSDL which Ant Script will use to generate complete WSDL file, these changes are in target "bindings" and "bindingsFactory"

```
< property name="binding.root" value ="HelloWorld5"/>
< property name="porttype.wsdl" value ="helloworld5_port_type.wsdl"/>

<property name="binding.root" value="FactoryService5"/>
<property name="porttype.wsdl" value="factory5_port_type.wsdl"/>
```

Two changes are related to the name of final WSDL file used to create stubs for both Services, these changes are in target "stubs" and "stubsFactory":

```
<property name="wsdl.file" value="HelloWorld5__service.wsdl"/>

<property name="wsdl.file" value="FactoryService5_service.wsdl"/>
```

Before we can run Ant script we have to put them at proper location where Ant build can locate them. Tutorial-5 is the main directory containing all files under one umbrella, our Web Service implementation *HelloWorldService5.java* will be in the directory structure according to package statement i.e. "*Tutorial-5\source\src\uk\ac\dl\ws\service*" our *Client5.java* will be the directory "*Tutorial-5\source\src\uk\ac\dl\ws\service\client*" (2). Our WSDL files "*helloworld5_port_type.wsdl*" and "*factory5_port_type.wsdl*" are located at "*Tutorial-5\source\schema\tutorial*" and source directory contains "deploy-server.wsdd", "deploy-jndi-config.xml" and "build.xml". File "post-deploy.xml", which; is used for creating client script is in directory "*Tutorial-5\source\etc*".

Compiling and Deploying Modified WSRF Web Service:

I am using Windows Operating System and have installed only WS-Core component of Globus Toolkit. You have to set an environment variable called GLOBUS_LOCATION referencing the installation of Globus Toolkit. In my case it is:

```
set GLOBUS_LOCATION= E:\GlobusWeek\gt-install
```

Open two Command Prompt one for starting Globus Tomcat Server and other for deploying Web Service and testing client. In one window go to directory *E:\GlobusWeek\ Tutorial-5\source* and run build file to create stubs, compile all java files, and deploy Web Service.

```
ant clean
ant deploy
```

If everything goes fine then after 16 seconds you will see the final outcome similar to:

```
generateWindows:
    [echo] Creating Windows launcher script helloworld5
    [copy] Copying 1 file to E:\GlobusWeek\gt-install\bin
   [delete] Deleting: E:\GlobusWeek\Tutorial-5\source\tmp\gar\etc\post-deploy.xml
```

Destroying the WSRF Resource (Immediate Destruction)

[delete] Deleting directory E:\GlobusWeek\Tutorial-5\source\tmp\gar

BUILD SUCCESSFUL
Total time: 16 seconds

In second Window go to directory "E:\GlobusWeek\gt-install" and run command:

bin\globus-start-container –nosec

This will start the container.

Testing Modified WSRF Web Service:

If you remember in the file "post-deploy.xml" we have mentioned helloworld5 as name of script to be generated to run the client, which is confirmed by the final outcome of the Ant build file. Now it is time to run the client:

%GLOBUS_LOCATION%\bin\helloworld5 -s http://localhost:8082/wsrf/services/HelloFactoryService5
new resource created...
Before Creating instance.
Created instance.
<ns1:name xmlns:ns1="http://tutorial3.wsrf.dl.ac.uk/helloworld">Asif Akram from Tutorial 5</ns1:name>
<ns1:name xmlns:ns1="http://tutorial3.wsrf.dl.ac.uk/helloworld">Rob Allan First Time Tutorial-5</ns1:name>
<ns1:name xmlns:ns1="http://tutorial3.wsrf.dl.ac.uk/helloworld">Rob Allan Second Time Tutorial-5</ns1:name>
Rob Allan Second Time Tutorial-5
Error: java.rmi.RemoteException: ; nested exception is:
org.globus.wsrf.NoSuchResourceException

"*Asif Akram from Tutorial 5*" is the initial value assigned to our variable "name" and "*Rob Allan First Time Tutorial-5*" is the value assigned to variable "name" from "Client5.java". Client5.java calls "echo" method twice and second time it passes the value "*Rob Allan Second Time Tutorial-5*". After this Client5.java calls Destroy() operation, there is no output for that operation call, but after destroying the resource; when client access the resource it has the exception "NoSuchResourceException".

Notice we are calling our HelloFactoryService5 and thorough HelloFactoryService5 we get the EndpointReferenceType of our HelloWorldService5. In the command prompt where we have started our container you can see both HelloFactoryService5 and HelloWorldService5 deployed. It is interesting to examine the SOAP messages to properly understand the inner working details of Factory Design Pattern. The first two SOAP messages are similar to what we have seen in the last tutorial, the only new thing is how Destroy request is sent and response to Destroy Request.

SOAP Request	Soap Response
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"> <soapenv:Header> <wsa:MessageID soapenv:mustUnderstand="0"> uuid:dcc1b900-f202-11d9-8c8b-ee769ff15e72 </wsa:MessageID> <wsa:To soapenv:mustUnderstand="0"> http://localhost:8082/wsrf/services/HelloFactoryService5 </wsa:To> <wsa:Action soapenv:mustUnderstand="0"> ***http://tutorial5.wsrf.dl.ac.uk/helloworld/HelloWorldFactoryPortType/createResourceRequest*** </wsa:Action> <wsa:From soapenv:mustUnderstand="0"> <wsa:Address> http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous </wsa:Address> </wsa:From>	<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"> <soapenv:Header> <wsa:MessageID soapenv:mustUnderstand="0"> uuid:dcd25ad0-f202-11d9-ac01-846fab68599d </wsa:MessageID> <wsa:To soapenv:mustUnderstand="0"> http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous </wsa:To> <wsa:Action soapenv:mustUnderstand="0"> http://tutorial5.wsrf.dl.ac.uk/helloworld/HelloWorldFactoryPortType/createResourceResponse </wsa:Action> <wsa:From soapenv:mustUnderstand="0"> <wsa:Address> http://localhost:8082/wsrf/services/HelloFactoryService5 </wsa:Address> </wsa:From>

Destroying the WSRF Resource (Immediate Destruction)

```
  </soapenv:Header>
  <soapenv:Body>
    <createResource
          xmlns="http://tutorial5.wsrf.dl.ac.uk/helloworld"/>
  </soapenv:Body>
</soapenv:Envelope>
```

```
    <wsa:RelatesTo RelationshipType="wsa:Reply"
      soapenv:mustUnderstand="0">
        uuid:dcc1b900-f202-11d9-8c8b-ee769ff15e72
    </wsa:RelatesTo>
  </soapenv:Header>
  <soapenv:Body>
    <createResourceResponse
        xmlns="http://tutorial5.wsrf.dl.ac.uk/helloworld">
      <wsa:EndpointReference>
        <wsa:Address>
          http://193.62.113.83:8080/wsrf/services/HelloWorldService5
        </wsa:Address>
        <wsa:ReferenceProperties>
          <NameKeyFactory5>9102746</NameKeyFactory5>
        </wsa:ReferenceProperties>
        <wsa:ReferenceParameters/>
      </wsa:EndpointReference>
    </createResourceResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Destroying the WSRF Resource (Immediate Destruction)

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:MessageID>
    <wsa:To soapenv:mustUnderstand="0">

http://193.62.113.83:8082/wsrf/services/HelloWorldService5
    </wsa:To>
    <wsa:Action soapenv:mustUnderstand="0">
      http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
      ResourceProperties/GetResourceProperty
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0">
      <wsa:Address>
     http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
      </wsa:Address>
    </wsa:From>
    <ns1: NameKeyFactory5
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
      soapenv:mustUnderstand="0">
          9102746
    </ns1: NameKeyFactory5 >
  </soapenv:Header>
  <soapenv:Body>
    <GetResourceProperty
       xmlns="http://docs.oasis-open.org/wsrf/2004/06/wsrf-
       WS-ResourceProperties-1.2-draft-01.xsd"
       xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
         ns1:name
     </GetResourceProperty>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:MessageID>
    <wsa:To soapenv:mustUnderstand="0">
       http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
    </wsa:To>
    <wsa:Action soapenv:mustUnderstand="0">
      http://docs.oasis-open.org/wsrf/2004/06/.....
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0"
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
      <wsa:Address>
        http://193.62.113.83:8082/wsrf/services/HelloWorldService5
      </wsa:Address>
      <wsa:ReferenceProperties>
        <ns1: NameKeyFactory5 soapenv:mustUnderstand="0"
          xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
          xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
            9102746
        </ns1: NameKeyFactory5 >
      </wsa:ReferenceProperties>
    </wsa:From>
    <wsa:RelatesTo RelationshipType="wsa:Reply"
      soapenv:mustUnderstand="0">
       uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:RelatesTo>
  </soapenv:Header>
  <soapenv:Body>
    <GetResourcePropertyResponse
      xmlns="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
       ResourceProperties-1.2-draft-01.xsd">
      <ns1:name xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
          Asif Akram from Tutorial 5
      </ns1:name>
    </GetResourcePropertyResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:3406ccc0-f2ba-11d9-9358-b9f80a10e3b4
    </wsa:MessageID>
    <wsa:To soapenv:mustUnderstand="0">

http://193.62.113.83:8082/wsrf/services/HelloWorldService5
    </wsa:To>
    <wsa:Action soapenv:mustUnderstand="0">
      http://.../wsrf-WS-ResourceLifetime/Destroy
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0">
      <wsa:Address>
        http://...../addressing/role/anonymous
      </wsa:Address>
    </wsa:From>
    <ns1:NameKeyFactory5
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
```

```
<soapenv:Envelope
   xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:34093dc0-f2ba-11d9-8a2d-f391724ecd56
    </wsa:MessageID>
    <wsa:To soapenv:mustUnderstand="0">
      http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
    </wsa:To>
    <wsa:Action soapenv:mustUnderstand="0">
      http://..../wsrf-WS-ResourceLifetime/DestroyResponse
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0"
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
      <wsa:Address>
        http://193.62.113.83:8082/wsrf/services/HelloWorldService5
      </wsa:Address>
      <wsa:ReferenceProperties>
        <ns1:NameKeyFactory5 soapenv:mustUnderstand="0"
          xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
          xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
```

Destroying the WSRF Resource (Immediate Destruction)

```
        soapenv:mustUnderstand="0">
          9102746
      </ns1:NameKeyFactory5>
    </soapenv:Header>
    <soapenv:Body>
      <Destroy
        xmlns="http://.../wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"/>
    </soapenv:Body>
</soapenv:Envelope>
```

```
          9102746
        </ns1:NameKeyFactory5>
      </wsa:ReferenceProperties>
    </wsa:From>
    <wsa:RelatesTo RelationshipType="wsa:Reply"
        soapenv:mustUnderstand="0">
          uuid:3406ccc0-f2ba-11d9-9358-b9f80a10e3b4
    </wsa:RelatesTo>
  </soapenv:Header>
  <soapenv:Body>
    <DestroyResponse
      xmlns="http://.../wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"/>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:MessageID>
    <wsa:To soapenv:mustUnderstand="0">

http://193.62.113.83:8082/wsrf/services/HelloWorldService5
    </wsa:To>
    <wsa:Action soapenv:mustUnderstand="0">
      http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
      ResourceProperties/GetResourceProperty
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0">
      <wsa:Address>
    http://schemas.xmlsoap.org/ws/2004/03/addressing/role/anonymous
      </wsa:Address>
    </wsa:From>
    <ns1:NameKeyFactory5
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
      soapenv:mustUnderstand="0">
        9102746
    </ns1:NameKeyFactory5>
  </soapenv:Header>
  <soapenv:Body>
    <GetResourceProperty
      xmlns="http://docs.oasis-open.org/wsrf/2004/06/wsrf-
      WS-ResourceProperties-1.2-draft-01.xsd"
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
        ns1:name
    </GetResourceProperty>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd=http://www.w3.org/2001/XMLSchema
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">
  <soapenv:Header>
    <wsa:MessageID soapenv:mustUnderstand="0">
      uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:MessageID>
    <wsa:Action soapenv:mustUnderstand="0">
      http://schemas.xmlsoap.org/ws/2004/03/addressing/fault
    </wsa:Action>
    <wsa:From soapenv:mustUnderstand="0"
      xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
      <wsa:Address>
        http://193.62.113.83:8082/wsrf/services/HelloWorldService5
      </wsa:Address>
      <wsa:ReferenceProperties>
        <ns1:NameKeyFactory5 soapenv:mustUnderstand="0"
          xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
          xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
          9102746
        </ns1:NameKeyFactory5>
      </wsa:ReferenceProperties>
    </wsa:From>
    <wsa:RelatesTo RelationshipType="wsa:Reply"
      soapenv:mustUnderstand="0">
      uuid:dd3b55d0-f202-11d9-8c8b-ee769ff15e72
    </wsa:RelatesTo>
  </soapenv:Header>
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode>soapenv:Server.userException</faultcode>
      <faultstring>java.rmi.RemoteException: ; nested exception is:
        org.globus.wsrf.NoSuchResourceException</faultstring>
      <detail>
        <ns1:stackTrace xmlns:ns1="http://xml.apache.org/axis/">
          java.rmi.RemoteException: ; nested exception is:
          org.globus.wsrf.NoSuchResourceException at
          HelloWorldService5.getResource(HelloWorldService5.java:34) at
          HelloWorldService5.getNameRP(HelloWorldService5.java:24) at
          ……
          HelloWorldService5.getResource(HelloWorldService5.java:32)
        </ns1:stackTrace>
        <ns2:hostname xmlns:ns2="http://xml.apache.org/axis/">
          AA78VIG
        </ns2:hostname>
      </detail>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

Let see SOAP messages to understand the working of Factory Design Pattern of WSRF. I have changed the port number of the service in the Client5.java to use TCP Monitor:

Destroying the WSRF Resource (Immediate Destruction)

```
// To Test SOAP messages in TCP Monitor
instanceEPR.getAddress().setPort(8082);
```

Call for CreateResource()

First SOAP request is calling business method createResource() of HelloFactoryService, but in response to this call WSRF HelloFactoryService is returning EndpointReferenceType with URL of HelloWorldService5.
SOAP Request:

```
<wsa:To soapenv:mustUnderstand="0">
    http://localhost:8082/wsrf/services/HelloFactoryService5
</wsa:To>
```

SOAP Response to createResource() operation:

```
<wsa:Address>
    http://193.62.113.83:8080/wsrf/services/HelloWorldService5
</wsa:Address>
<wsa:ReferenceProperties>
    < NameKeyFactory5 >9102746</ NameKeyFactory5 >
</wsa:ReferenceProperties>
<wsa:ReferenceParameters/>
```

If you remember from previous tutorials EndpointReferenceType is wrapper around WSRF Service URL and unique ID. The name of unique ID is NameKeyFactory5, which was mentioned in the deploy-jndi.xml as

```
<parameter>
    <name>resourceKeyName</name>
    <value>{http://tutorial5.wsrf.dl.ac.uk/helloworld}NameKeyFactory5</value>
</parameter>
```

Call for GetResourceProperty ()

For further interaction with WSRF service client has to send EndpointReferenceType to interact with the same instance as it is done in the second SOAP request, EndpointReferenceType information is sent in the SOAP header.

```
<wsa:To soapenv:mustUnderstand="0">
    http://193.62.113.83:8082/wsrf/services/HelloWorldService5
</wsa:To>
<ns1:NameKeyFactory5 xmlns:ns1="http://tutorial4.wsrf.dl.ac.uk/helloworld" soapenv:mustUnderstand="0">
    9102746
</ns1: NameKeyFactory5 >
```

In response to second SOAP request, WSRF Service is sending EndpointReferenceType in the SOAP Response header and result of the called business method is in the body of SOAP Response.

```
<soapenv:Header>
    …..
    …..
    <wsa:From soapenv:mustUnderstand="0" xmlns:ns1="http://tutorial3.wsrf.dl.ac.uk/helloworld">
      <wsa:Address>
        http://193.62.113.83:8082/wsrf/services/HelloWorldService5
      </wsa:Address>
      <wsa:ReferenceProperties>
        <ns1: NameKeyFactory5 soapenv:mustUnderstand="0"
          xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld"
          xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
            9102746
```

Destroying the WSRF Resource (Immediate Destruction)

```
        </ns1: NameKeyFactory5 >
      </wsa:ReferenceProperties>
    </wsa:From>
    …..
    …..
  </soapenv:Header>
  <soapenv:Body>
    <GetResourcePropertyResponse
      xmlns="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd">
      <ns1:name xmlns:ns1="http://tutorial5.wsrf.dl.ac.uk/helloworld">
          Asif Akram from Tutorial 5
      </ns1:name>
    </GetResourcePropertyResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Each client will have unique ID of its resource, and if two clients have same unique ID then it means that they are sharing the same instance of the resource.

Call for Destroy ()

Now Client5.java calls the Destroy() operation, which is very simple and SOAP Response to this request is empty, there is no true/false Boolean value depending on if destroy operation is successful or not.

SOAP Request for Destroy() operation:

```
<soapenv:Body>
  <Destroy xmlns="http://./wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"/>
</soapenv:Body>
```

SOAP Response for Destroy() operation:

```
<soapenv:Body>
  <DestroyResponse  xmlns="http://.../wsrf-WS-ResourceLifetime-1.2-draft-01.xsd"/>
</soapenv:Body>
```

Second Call for GetResourceProperty() after Destroy()

SOAP Request for GetResourceProperty() after the destruction of the resource is very similar, to the first call, but SOAP Response is very different, below is the part of SOAP Response:

```
<soapenv:Body>
  <soapenv:Fault>
    <faultcode>soapenv:Server.userException</faultcode>
    <faultstring>java.rmi.RemoteException: ; nested exception is:
      org.globus.wsrf.NoSuchResourceException</faultstring>
    <detail>
      <ns1:stackTrace xmlns:ns1="http://xml.apache.org/axis/">java.rmi.RemoteException: ;
        nested exception is:  org.globus.wsrf.NoSuchResourceException
        at uk.ac.dl.ws.service.HelloWorldService5.getResource(HelloWorldService5.java:34)
        at uk.ac.dl.ws.service.HelloWorldService5.getNameRP(HelloWorldService5.java:24)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:324)
```

Destroying the WSRF Resource (Immediate Destruction)

```
        at org.apache.axis.providers.java.RPCProvider.invokeMethod(RPCProvider.java:384)
        at org.globus.axis.providers.RPCProvider.invokeMethodSub(RPCProvider.java:104)
        at org.globus.axis.providers.RPCProvider.invokeMethod(RPCProvider.java:87)
        at org.apache.axis.providers.java.RPCProvider.processMessage(RPCProvider.java:281)
        at org.apache.axis.providers.java.JavaProvider.invoke(JavaProvider.java:319)
        at org.apache.axis.strategies.InvocationStrategy.visit(InvocationStrategy.java:32)
        at org.apache.axis.SimpleChain.doVisiting(SimpleChain.java:118)
        at org.apache.axis.SimpleChain.invoke(SimpleChain.java:83)
        at org.apache.axis.handlers.soap.SOAPService.invoke(SOAPService.java:450)
        at org.apache.axis.server.AxisServer.invoke(AxisServer.java:285)
        at org.globus.wsrf.container.ServiceThread.doPost(ServiceThread.java:662)
        at org.globus.wsrf.container.ServiceThread.process(ServiceThread.java:393)
        at org.globus.wsrf.container.ServiceThread.run(ServiceThread.java:297)
        Caused by: org.globus.wsrf.NoSuchResourceException
        at org.globus.wsrf.impl.ResourceHomeImpl.get(ResourceHomeImpl.java:281)
        at org.globus.wsrf.impl.ResourceHomeImpl.find(ResourceHomeImpl.java:255)
        at org.globus.wsrf.impl.ResourceContextImpl.getResource(ResourceContextImpl.java:161)
        at uk.ac.dl.ws.service.HelloWorldService5.getResource(HelloWorldService5.java:32)
        ... 18 more
      </ns1:stackTrace>
      <ns2:hostname xmlns:ns2="http://xml.apache.org/axis/">AA78VIG</ns2:hostname>
    </detail>
  </soapenv:Fault>
</soapenv:Body>
```

CCLRC

Destroying the WSRF Resource (Immediate Destruction)